WHFS Alert/Alarm Operations Guide

National Weather Service
Office Of Hydrologic Development
February 12, 2002

Table of Contents

1. INTRODUCTION

Beginning with Release 5.0 of AWIPS, the Integrated Hydrologic Forecast System (IHFS) database structure and the WFO Hydrologic Forecast System (WHFS) applications have included an Alert/Alarm capability. This Alert/Alarm functionality can be broken into the following four main components:

1) Functionality that allows the user to set the alert and alarm thresholds.

2) Data processing that identifies and stores data that exceeds these alert or alarm thresholds.

3) Functionality to view the stored alert and alarm data in a summary tabular format and graphically within it timeseries context.

4) Capability to report selected alert/alarm data in a product. This product can be sent to the text database and can be viewed or disseminated as desired.

Note:
In the WHFS there is processing for monitoring data that may exceed some alert or alarm thresholds and there is processing related to quality control. Operations for both of these features require an application to compare the data value(s) with some predefined threshold. When processing data, it makes sense from a speed/performance standpoint to perform the alert/alarm and QC operations together. Both require the physical data and the threshold limits data to be read and compared. In practice, this takes an amount of time which is time best spent if done simultaneously. Even though some of the functionality mentioned later are also performing QC checking, this document only describes the alert/alarm operations. The QC operations are described in a separate document. This document describes the operational aspects of the four main components of the Alert/Alarm operations detailed above.

2. DESCRIPTION OF ALERT/ALARM OPERATIONS

2.1 Setting Alert and Alarm Thresholds

In WHFS release 5.0, a new dialog, Quality Control and Alert/Alarm Limits, was added to the HydroBase Database Manager. This dialog allows the user to set alert and alarm thresholds for any physical element at any location. A view of this dialog display is seen in Appendix B.

The user has the option to set general limits for a physical element (PE) that apply to all locations (Default Limits) or Location Limits that apply only to a specific location, physical element, duration, and time of year. Example displays of both Default and Location Limits can be seen in Appendix B. These user defined limits are stored in two tables in the IHFS database. The limits associated with a specific location are stored in the LocDataLimits table, while the Default Limits are stored in the DataLimits table. If a Location Limit is set for a particular location and physical element, that threshold overrides the Default limit for that physical element.

In addition, the user can set thresholds for both a single value and for rate of change (ROC). The single value limit is simply a limit that is applied to all incoming hydrometeorologic data being processed by the SHEF Decoder. The rate of change limits refer to an unit/hour limit that is applied to data in specified observed value tables in the IHFS database.   The user can establish single value and ROC limits for two separate categories, Alert and Alarm, with Alert being the less serious of the two.

2.2 Data Processing to Identify Data that Exceeds Limits

As described above, the user sets two types of limits, single value limits and rate of change (ROC) limits as part of the Alert/Alarm operations. The following describes the processes that handle the checking of data against these two types of limits:

2.2.1 Single Value Checks

Single value checks are performed by the SHEF Decoder against all incoming hydrometeorologic data. Assuming the Apps_defaults token is set to ON, (shef_alertalarm = ON), the SHEF Decoder checks the incoming data against the user defined alert and alarm single value thresholds. The limit that is applied to each piece of incoming SHEF data depends on the data that is in the LocDataLimits and DataLimits tables described above.

As the data is processed by the SHEF Decoder, the LocDataLimits is searched for limits specified for the data value's location, physical element, duration, and time. If a location-specific limit is found in this table, then it is used in the single value check performed by the SHEF Decoder.

If a specific location limit is not found, then the more general set of limits stored in the DataLimits table is searched. This table is searched to see whether it has limits for the data value's physical element, duration, and time. The location is not considered when searching in this table. If an entry for the data value's physical element, duration, and time is found in the DataLimits table, then the location-independent general limit is used. The effect is that the LocDataLimits table allows the user to override the general data limits for specific locations.

If neither a location-dependent (i.e. from LocDataLimits) nor a location-independent (i.e. from DataLimits) is found, then no limits are available and the limit tests are not performed. For the searches of both tables, note that the limits are given not just for the physical element, but also for the duration, and for the time of year. The time of year restriction allows seasonally-based limits to be defined.

In these two tables, a limit is assumed to be defined if a value is specified, as opposed to a “null”, or blank, value being specified. Therefore, setting a limit to 0.0 does not deactivate the relevant test. To turn off a given test, a blank value must be specified which results in a null value being inserted into the database.

Note that when obtaining the limits for a given location, the applications will only use the limits in one of the two tables. If a user defines a given test’s limit for a specific location, then a record will be created for that location in the LocDataLimits table, and that record may contain values for all limits, or for only some limits. Anytime an application needs a limit, it first looks in the location-specific record for that location. If the location is defined, but only some of its limits are specified, then for the other tests, it will not find the limit value. In this case, the application will NOT attempt to search the general location-independent limits for the same physical element, duration, and time of year. This allows the user to avoid performing these other tests for a location, while still performing certain tests.

Also note that changes to the single-value checks will not take affect until the SHEF decoder application is restarted. This is because the SHEF decoder application buffers up the limit values in order to speed processing. However, changes to the rate-of-change thresholds will take effect the next time the rate-of-change checker application is executed.

2.2.2 Rate of Change (ROC) Checks

A scheduled cron job is provided as part of the standard WHFS installation to perform automated checks on the data. This mode of checking is most relevant for checks which require multiple-values, whether they be values for multiple times for one location, or for multiple locations for one time. To do these checks as each value is posted would not make sense because the necessary data would generally not be available. Also, it would be extremely burdensome because of the typically large throughput of data entering the IHFS database. These automated checks include the ROC checks that are part of the alert/alarm operations.

A timed cron job approach is used to allow the data to be posted and build up to a level that provides enough values to properly perform the ROC checks. The cron job that is executed is the script:

    /awips/hydroapps/whfs/bin/run_alarm_whfs

This script operates by actually running two other scripts. The first performs the automated checking of the data, for both alert/alarm monitoring and quality control. The second is strictly for alert/alarm purposes; it generates and distributes a report of the current alert/alarm conditions and will be described later in this document. The ROC checks are performed by the first script it executes:

/awips/hydroapps/whfs/bin/run_roc_checker

This job performs rate-of-change checks for the observed physical element tables specified in the script, and is run at a specified frequency. The default setting is to run the checks on the Height and Precipitation (type-source of PC data only) tables every 10 minutes, and considering the data for the previous 6 hours. The script invokes the roc_checker program for one table per execution. Only data in the observed physical element tables are supported. In addition to the command-line arguments supplying the database name and the physical element table to process, the rate-of-change checker program supports optional arguments. These include:

1) an endtime as a relative or absolute time; default is now
2) the number of hours to process; default is 6 hours
3) a specific location to process; default is all locations
4) a specific physical element to process; default is all physical elements within the given physical element table
5) log error messages only; default is to show all
6) use GOOD data, or GOOD and QUESTIONABLE data; default is to use GOOD data only

The text of the script that specifically describes how to call the roc_checker program including the required and optional arguments contains the following:

```
# program usage:
#
# roc_checker -d<database> -t<table>   <--required args
#  database          : name of database, supplied by script
#  table             : name of the table whose data will be reviewed.  The names are
#                      one of the following tables in the Informix database which
#                      contain observed data:  agricultural, discharge,
#                      evaporation, gatedam, ground, height, ice, lake, moisture, precip,
#                      pressure, procvalue, radiation, snow, temperature, weather,
#                      wind, yunique
# optional args:
#  -h<endtime>       : default=current system time; can specify time as 1-4 digit
#                      number which is the numbers of hours preceding the current
#                      time, or an absolute time can be specified in the format
```

```
#                        yyyymmddhh.
#  -n<numhrs>            : default = 6 hrs; defines the starting time as the given
#                        number of hours before the ending time
#  -l<lid>               : default = all stations; can specify single station id
#  -p<pe>                : default = all SHEF pe codes; can specify single SHEF pe code
#  -err                  : default = show all messages; if arg specified, only err
#                        msgs written
#  -u<G | GQ>            : type of data used in quality code checks.
#                        'G' denotes the use of "good" data;
#                        'GQ' denotes the use of "good questionable" data;
#                        default = 'G' ("good data").

Dte=`date -u`
echo Calling roc_checker$OS_SUFFIX at  $Dte >> $LOGFILE

$WHFS_BIN_DIR/roc_checker$OS_SUFFIX -d$DB_NAME -theight        >> $LOGFILE
$WHFS_BIN_DIR/roc_checker$OS_SUFFIX -d$DB_NAME -tprecip -pPC >>$LOGFILE
```

The roc_checker checks data for the specified table against the alert and alarm ROC limits found in the LocDataLimits and DataLimits tables. Just as in the single value checks, the location specific limits in LocDataLimits take precedence over the general ROC limits in the DataLimits table.

2.2.3 Storage of Data

The normal destination for data processed by the SHEF decoder is a set of tables referred to as the physical element (PE) tables. These are the primary tables in which all operational hydrometeorological data are stored. As data are identified that exceeds the applied alert/alarm thresholds in the single value checks by the SHEF Decoder or the ROC checks executed on the CRON, a copy of the data are stored in the AlertAlarmVal table in the IHFS database. The data stored in the AlertAlarmVal table has all the attributes of the data stored in the PE tables, but also specifies whether the data value exceed the alert or the alarm limitation and whether it was a single value or rate of change exceedence. These data are generally kept in the IHFS database for a short period (nominally 12 hours), but this is configurable.

This method of storing a copy of the data that exceeds the applied alert/alarm thresholds in a separate AlertAlarmVal table is much different than the method used to store/identify data that exceeds QC limits. Data that exceeds QC limits are not copied to a separate table. Instead, they are only stored in one table (either the PE table or the RejectedData table per the IHFS token, shef_post_baddata, controlled by the user). QC data are identified by the value of two fields (shef_qual_code and quality_code) within the PE tables.

2.3 Viewing the Alert/Alarm Data

The user can view the data that have exceeded alert and alarm thresholds based on value and rate-of-change utilizing the HydroView Data Viewer. A new Alert and Alarm Data dialog allows the user to view this data. The user can sort data by time or location. The user can filter the data by observation/forecast/both, alert/alarm/both,and value/rate-of-change/both. The user-set alert/alarm limits are provided for reference and the user can jump directly to the Time Series functionality to view the data within its time series context easily. Several examples of the displays Alert/Alarm displays in Hydroview are included in Appendix B.

2.4 Reporting the Alert/Alarm Data

As discussed in 2.2.2 above, two scripts are called by the cron job executed by the script:

/awips/hydroapps/whfs/bin/run_alarm_whfs

The first script called is the run_roc_checker. Once the roc_checker has finished, the second script that is executed is:

/awips/hydroapps/whfs/bin/run_report_alarm

This script generates a report of the current alert/alarm conditions and distributes this report to the text database. Although, the basic function of the report_alarm is simply to query the data exceeding alert/alarm thresholds stored in the AlertAlarmVal table and produce a report, the run_report_alarm script is highly configurable by the user. The basic configurability of the reporting is described in the run_report_alarm script as follows:

```
# program usage:
#
# report_alarm -d<database> -p<product_id>                  <--required args
#            -r<report_mode> -s<file_suffix> -m<minutes>    <-- optional args
#            -f<generalfilter>  -e<PEfilter>               <-- optional args
#
#  database    : name of database, supplied by script
#  product_id  : nine-character product_id, used to set the output filename
#               and for sending the product to the text database
#               !!!!!!  THE LOCAL OFFICE SHOULD SET THIS ACCORDINGLY !!!!!!!!
#
# optional args:
#
#  filter      : set of characters which instruct the program to only consider
```

```
#             certain types of alert/alarms for reporting.
#    O - Read observed alert/alarms, which has a type-source of either R* or P*
#    F - Read forecast alert/alarms, which has a type-source of either F* or C*
#    R - Read exceed rate-of-change alert/alarms.
#    V - Read exceed value alert/alarms.
#    M - Read alarm data.
#    T - Read alert data.
#    e.g.: -fORT == read observed rate-of-change alerts only.
#             Note that one can specify O and F, or R and V, or M and T together,
#             but that is equivalent to specifying nothing since the default is
#             to include both observed and forecast alert/alarms, both rate-of-change
#             and value exceed alerts/alarms, and both alerts and alarms
#             (i.e. the default is -fOFRVMT).
#
#  report_mode : reporting mode, set to one of the following:
#    ALL -       Report all records - and that means ALL records.
#
#    RECENT -     Report all observed or forecast records posted within the
#              past xxx minutes.
#
#    NEAR_NOW -   Reported observed values within the past xxx minutes and
#              forecast values within the next xxx minutes.
#
#    NEAREST -    Reporting most recent observed value and earliest
#              forecast value.
#
#    LATEST_MAXFCST - Reporting most recent observed value and maximum
#                forecast value.
#
#    UNREPORTED - Report all records that were not previously reported,
#              as noted by the action_time field being not equal to null.
#
#    NEAREST - Report the most recent record for observed data, or the earliest
#             record for forecast data, for each "group" of data, where a
#             group is for a unique lid/pe/ts.  This is done regardless of
#             whether it has been previously reported.
#
#    FRESH -  For observed data, listing all records that are later than xxx
#            minutes after the time of the most recent reported value.
#            For forecast data, listing the maximum forecast value if it
#            was not reported within the past xxx minutes.
#
#    NEW_OR_INCREASED - For observed data, listing unreported records if,
#            the previous reported was later than xxx minutes ago OR if the
#           report has a higher value than the last reported record.
```

```
#           For forecasted data, listing the maximum forecast value if it
#           was not reported within the past xxx minutes.
#
#   minutes    : the number of minutes used for time window purposes in
#             the reporting modes: RECENT, NEAR_NOW, FRESH,
NEW_OR_INCREASED.
#
#  file_suffix : string appended to the end of the product id to form
#             the output file name.  the file suffix is set by this script
#             to be based on the system time.
#
# - Whenever a record is reported, the action_time field is updated!!!
# - Regardless of the report_mode, the data are always filtered by
#   and filter options; i.e. the filter is applied FIRST!!!
#
#   ALL and UNREPORTED are very simple modes; all mean all and
#   unreported means unreported.  These modes are handled easily
#   by filtering the retrieved data when it is read from the database.
#
#   RECENT and NEAR_NOW are also simple modes.  RECENT means
#   recently posted, NEAR_NOW means observed or forecast data
#   near now.
#
#   NEAREST is also pretty simple, in that it reports the nearest
#   observed of forecast data, in terms of time.
#
#   LATEST_MAXFCST is also straightforward in that it reports the
#   latest (i.e. nearest) observed data, but unlike the NEAREST mode,
#   reports the maximum forecast value.
#
#   FRESH and NEW_OR_INCREASED use the most complex concepts.
#
#   The following modes do not consider the action_time:
#   ALL, RECENT, NEAR_NOW, NEAREST, LATEST_MAXFCST
#   These other modes do consider the action_time:
#   UNREPORTED, FRESH, NEW_OR_INCREASED
```

Appendix A. Diagram of Alert/Alarm Operations in WHFS

Appendix B. Example Displays from Alert/Alarm Applications